

**ІНФОРМАТИКА, КОМП'ЮТЕРНА ТЕХНІКА**

УДК 004.056.53

**БЕЗПЕКА РОЗПОДІЛЕНИХ ЗАСТОСУНКІВ, ПОБУДОВАНИХ НА ОСНОВІ  
SPRING CLOUD**

**А. Джоші**

студент 5 курсу, група КНІТ-51м, навчально-науковий інститут автоматики, кібернетики та  
обчислювальної техніки

Науковий керівник – д.т.н., професор П. М. Мартинюк

*Національний університет водного господарства та природокористування,  
м. Рівне, Україна*

**В статті наведено результати аналізу захищеності розподілених застосунків, побудованих на основі Spring Cloud, виявлено вразливості та надано рекомендації, що допоможуть уникнути або зменшити ризик проведення атак.**

**Ключові слова:** розподілений застосунок, фреймворк, захищеність, сервер.

**В статье представлены результаты анализа защищенности распределенных приложений, построенных на основе Spring Cloud, обнаружены уязвимые места и даны рекомендации, которые помогут избежать или уменьшить риск проведения атак.**

**Ключевые слова:** распределенное приложение, фреймворк, защищенность, сервер.

**In the article analysis of distributed applications security has been done, vulnerabilities were found and practical recommendations that should reduce or avoid risk of attacks appearance were shown.**

**Keywords:** distributed application, framework, security, server.

**Останнім часом питання** продуктивності серверних застосунків займає все важливіше місце в світі розробки програмного забезпечення. Збільшення кількості клієнтів, ускладнення логіки застосунку, множинні сервери баз даних – всі ці фактори впливають на розвиток розподілених застосунків. Розподілені застосунки мають низку переваг над монолітними застосунками в контексті продуктивності: швидкість, за рахунок більшої кількості фізичних серверів; відмовостійкість – при раптовій зупинці одного з серверів, застосунок загалом продовжуватиме роботу.

Саме тому розробники програмного забезпечення все частіше надають перевагу розподіленим застосункам, якщо до системи поставлені конкретні вимоги щодо продуктивності та відмовостійкості. Попит на розподілені застосунки став рушійною силою розвитку екосистеми розподілених застосунків, яка складається з низки альтернативних технологій та фреймворків. На даний час найпопулярнішими фреймворками для побудови розподілених застосунків є: Microsoft.net remoting, розподілені застосунки на основі node.js, розподілені застосунки на основі інфраструктури Apache Software Foundation та застосунки побудовані на основі Spring Cloud.

**Розподілений застосунок** – застосунок, який складається з набору незалежних об'єктів і подається користувачу як єдина об'єднана система [1]. Призначення розподіленого застосунку полягає в забезпеченні загальної працездатності застосунку, що складається із різних об'єктів. Розподілені застосунки не надають додаткових функцій моніторингу для об'єктів в такому застосунку. Натомість вони включають об'єкти, які вже знаходяться під

наглядом. При розробці розподіленого застосунку основною проблемою є створення зв'язку між об'єктами, що є частинами одного застосунку [2].

Питанням безпеки розподілених застосунків присвячені праці таких науковців, як Ендрю Таненбаум, Маартен ван Стен, Кенні Бастані, Джоша Лонга та інші.

**Методика досліджень.** Для дослідження проблеми захищеності розподілених застосунків, побудованих на основі Spring Cloud було обрано постановку експерименту щодо виявлення вразливостей прикладного застосунку та метод спостереження за поведінкою застосунку під час проведення атак.

**Для проведення аналізу** захищеності застосунків, побудованих на основі Spring Cloud було поставлено наступні завдання: огляд теоретичних відомостей з розподілених застосунків; побудова розподіленого застосунку на основі Spring Cloud; проведення атак на даний застосунок; наведення практичних рекомендацій щодо покращення рівня захищеності застосунку.

**Сучасні фреймворки** для побудови розподілених застосунків відрізняються спрощеною моделлю розробки, розгортки та управління застосунком. Spring Cloud – бібліотека з відкритим кодом, що спрощує створення JVM розподілених застосунків. Використання даної бібліотеки дозволяє спростити підключення інфраструктурних сервісів, таких як джерело даних (data source), менеджер транзакцій, менеджер безпеки тощо.

До фреймворку Spring Cloud входять й інфраструктурні компоненти для побудови розподілених застосунків. Одним із найважливіших компонент є сервер конфігурацій застосунку. Сервер конфігурацій є централізованим сховищем конфігурацій для всіх вузлів застосунку, використання якого значно спрощує розробку та підтримку розподіленого застосунку. Служба знаходження сервісів є зручним компонентом автоматичного пошуку сервісів, що належать певному розподіленому застосунку. Важливою частиною фреймворку, та будь-якого розподіленого застосунку є балансир навантаження, призначенням якого є рівномірний розподіл навантаження на всі вузли застосунку, що підвищує загальну потужність застосунку для великої кількості запитів та даних, що оброблюються. Шлюз інтерфейсу прикладного програмування використовується як централізована точка входу для усіх клієнтських застосунків, що обслуговуються розподіленим серверним застосунком. Широкого застосування отримали й сервіси відмовостійкості, що займаються приховуванням помилок при некоректному використанні кінцевих точок серверних вузлів застосунку. Сервіси відмовостійкості, в тому числі мають ряд вбудованих веб-застосунків, призначення яких полягає у моніторингу стану застосунку. Архітектура типового розподіленого застосунку побудованому на Spring Cloud зображена на рис. 1.

Для проведення аналізу захищеності розподілених застосунків було побудовано відповідний приклад та проведено ряд атак на нього. Важливою властивістю розподілених застосунків є протистояння атакам типу «відмова від обслуговування». В ході виконання даної атаки було визначено, що сервер ніяким чином не блокує даний потік запитів та намагається обробити кожен з них. Після тривалого напливу запитів, продуктивність сервера різко падає. Швидкість обробки запиту коливається від максимально можливої для даного середовища запуску до нульової. Внаслідок проведення стресового тестування даного застосунку було отримано результати про відсутність протидії до атак типу «відмова у обслуговуванні». Для захисту застосунку від атак типу «відмова в обслуговуванні» рекомендується використання сторонніх сервісів. Найчастіше для захисту розподілених застосунків використовуються можливості середовища розгортання застосунку, таких як Amazon Shield.

Коректність роботи каналу повідомлень неможлива без перевірки відправника та отримувача повідомлень. Для аналізу захищеності застосунку від підміни повідомлень було створено сторонній застосунок відправки повідомлення на задану підписку. Після відправки повідомленні зі стороннього застосунку, повідомлення було поміщене до черги брокера

повідомлень після чого було успішно оброблене сервісом-обробником. Саме по причині неможливості ідентифікації відправника в широкомовному брокері повідомлень рекомендується використовувати захищені черги повідомлень або перевірку відправника в заголовку повідомлення на стороні вузлів-обробників.

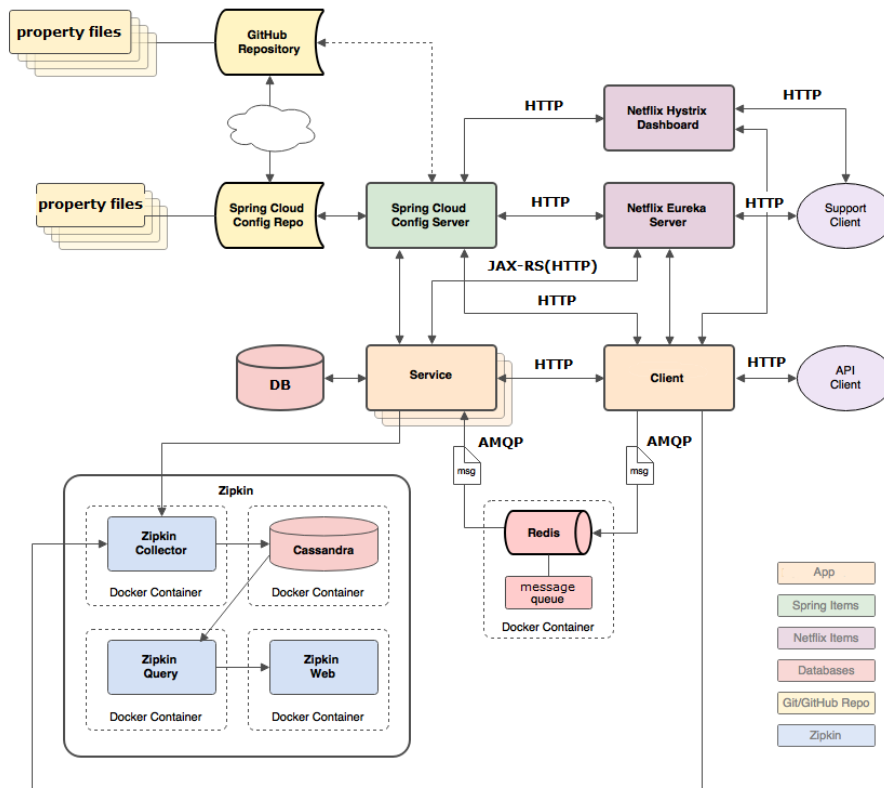


Рис. 1. Архітектура типового Spring Cloud застосунку

В даному розподіленому застосунку існує зв'язок між проксі-шлюзом та сервісами-обробниками запитів. При виконанні запиту на проксі-шлюз виконується його переадресація на сервіс-обробник. При аналізі трафіку було виявлено, що в якості протоколу зв'язку між ними використовується HTTP. В результаті перехоплення мережевого трафіку було отримано всі відповіді сервісу-обробника в незашифрованому форматі (рис. 2).

Packet	Hostname	Content Type	Size	Filename
14	localhost:8000	application/json	35 bytes	reservations
32	localhost:8000	application/hal+json	265 bytes	reservations
98	localhost:8761	application/json	82 bytes	delta
123	localhost:8761	application/json	82 bytes	delta
164	localhost:8761	application/json	82 bytes	delta
174	localhost:8000	application/json	37 bytes	reservations
192	localhost:8000	application/hal+json	267 bytes	reservations
303	localhost:8000	application/json	39 bytes	reservations
334	localhost:8761	application/json	82 bytes	delta
348	localhost:8000	application/hal+json	271 bytes	reservations
385	localhost:8761	application/json	82 bytes	delta
422	localhost:8761	application/json	82 bytes	delta
521	localhost:8000	application/json	39 bytes	reservations
541	localhost:8000	application/hal+json	271 bytes	reservations
598	localhost:8761	application/json	82 bytes	delta
623	localhost:8761	application/json	82 bytes	delta
680	localhost:8761	application/json	82 bytes	delta
698	localhost:8000	application/json	39 bytes	reservations
710	localhost:8000	application/hal+json	271 bytes	reservations
784	localhost:8000	application/hal+json	4168 bytes	reservations
800	localhost:8080	application/hal+json	4168 bytes	reservations
853	localhost:8761	application/json	82 bytes	delta
880	localhost:8761	application/json	82 bytes	delta
917	localhost:8761	application/json	82 bytes	delta

Рис. 2. Відповіді сервісу-обробника

Тому для зв'язку між вузлами розподіленого застосунку рекомендується використовувати зв'язок за допомогою обміну повідомлень або використання криптографічних фільтрів на вході та виході вузла.

Передача повідомлень є однією з найважливіших функцій розподілених застосунків. Однією з основних функцій застосунку є безпека каналу передачі таких повідомлень. Для перевірки захищеності каналу було проведено аналіз мережевого трафіку на вузлах застосунку. Оскільки в якості брокера повідомлень було обрано RabbitMQ – фільтром протоколу передачі повідомлень є AMQP. З результату датаграми видно, що протокол AMQP використовується лише в цілях сповіщення кінцевих точок каналу про працездатність каналу передачі повідомлень, але не є способом передачі самих повідомлень. Після подальшого аналізу TCP-трафіку не було виявлено відкритої передачі повідомлення від однієї кінцевої точки до іншої, тому передачу повідомлень можна вважати захищеною.

В ході аналізу TCP-трафіку було виявлено фрагмент датаграми (рис. 3), що відповідає heartbeat-паketу проксі-шлюзу застосунку. В даному пакеті в незашифрованому вигляді знаходиться ідентифікатор застосунку та ідентифікатор вузла, за яким у злоумисника при наявності інтерфейсу взаємодії з сервером пошуку вузлів є можливість відключення знайденого вузла від застосунку шляхом виконання DELETE запиту. Дереєстрація проксі-шлюзу застосунку спричиняє неможливість обробки будь-яких вхідних запитів, що є різновидом атаки виду «відмова в обслуговуванні». Для захисту застосунку від подібних атак рекомендується вимкнення публічного інтерфейсу сервера пошуку вузлів та закриття зайвих портів на машинах розгортання застосунку.

The screenshot shows a network traffic analysis tool interface. The top part displays a list of captured packets with columns for No., Time, Source, Destination, Protocol, Length, and Info. The bottom part shows a detailed view of a selected packet's header and payload.

No.	Time	Source	Destination	Protocol	Length	Info
103	8.581766	127.0.0.1	127.0.0.1	TCP	40	8761->62194 [ACK] Seq=1 Ack=278 Win=525568 Len=0
104	8.583767	127.0.0.1	127.0.0.1	HTTP	256	HTTP/1.1 200 (application/json)
105	8.583767	127.0.0.1	127.0.0.1	TCP	40	62194->8761 [ACK] Seq=278 Ack=217 Win=525312 Len=0
106	8.583767	127.0.0.1	127.0.0.1	TCP	41	61422->61423 [PSH, ACK] Seq=1 Ack=1 Win=2053 Len=1
107	8.583767	127.0.0.1	127.0.0.1	TCP	40	61423->61422 [ACK] Seq=1 Ack=2 Win=2051 Len=0
108	8.584768	127.0.0.1	127.0.0.1	TCP	41	61426->61427 [PSH, ACK] Seq=2 Ack=1 Win=2053 Len=1
109	8.584768	127.0.0.1	127.0.0.1	TCP	40	61427->61426 [ACK] Seq=1 Ack=3 Win=2051 Len=0
110	8.585769	127.0.0.1	127.0.0.1	TCP	52	62195->8761 [SYN] Seq=0 Win=0 Len=0 MSS=65495 WS=256 SACK_PERM=1
111	8.585769	127.0.0.1	127.0.0.1	TCP	52	8761->62195 [SYN, ACK] Seq=0 Ack=1 Win=0 Len=0 MSS=65495 WS=256 SACK_PERM=1
112	8.585769	127.0.0.1	127.0.0.1	TCP	40	62195->8761 [ACK] Seq=1 Ack=1 Win=525568 Len=0
113	8.585769	127.0.0.1	127.0.0.1	TCP	41	61424->61425 [PSH, ACK] Seq=1 Ack=1 Win=2053 Len=1
114	8.585769	127.0.0.1	127.0.0.1	TCP	40	61425->61424 [ACK] Seq=1 Ack=2 Win=2051 Len=0
115	8.586769	127.0.0.1	127.0.0.1	HTTP	395	PUT /eureka/apps/RESERVATION-CLIENT/localhost:reservation-client?status=UP&lastDirtyTimestamp=1496083781556
116	8.586769	127.0.0.1	127.0.0.1	TCP	40	8761->62195 [ACK] Seq=1 Ack=356 Win=525568 Len=0

Header Length: 20 bytes  
 Flags: 0x018 (PSH, ACK)  
 000. .... = Reserved: Not set  
 ...0. .... = Nonce: Not set  
 ....0. .... = Congestion Window Reduced (CWR): Not set  
 ....0. .... = ECH-Echo: Not set  
 ....0. .... = Urgent: Not set  
 ....1. .... = Acknowledgment: Set  
 ....1. .... = Push: Set  
 ....0. .... = Reset: Not set  
 ....0. .... = Syn: Not set  
 ....0. .... = Fin: Not set

0000 45 00 01 0b 2b 21 40 00 80 06 00 00 7f 00 00 01 E...+!@.....  
 0010 7f 00 00 01 f2 f3 22 39 8d af f6 80 1f 94 7e b2 .....9.....  
 0020 00 05 5a 9f 00 00 50 55 54 20 2f 65 75 72 00 00 .2... PUT /eur  
 0030 65 60 61 2f 61 70 70 73 2f 52 45 53 45 52 56 41 aka/apps/RESERVA  
 0040 54 49 4f 4e 2d 43 4c 49 45 4e 54 2f 6c 6f 63 61 TION-CLI ENT/loca  
 0050 6c 68 6f 73 74 3a 72 65 73 65 72 76 61 74 69 6f lhost:re servatio  
 0060 66 2d 63 6c 69 65 6e 74 3f 73 74 61 74 75 73 3d n-client ?status=  
 0070 55 50 26 6c 61 73 74 44 69 72 74 79 54 69 6d 65 UP&last DirtyTime  
 0080 73 74 61 6d 70 3d 31 34 39 36 30 38 33 37 38 31 stamp=14 96083781

Рис. 3. Фрагмент датаграми

**Висновки.** Отже, проблемами застосунків, побудованих на основі Spring Cloud є неможливість протидії атакам по типу «відмова в обслуговуванні», передача незашифрованої інформації по незахищеному каналу передачі даних, можливість підміни повідомлень, перехоплення та використання службової інформації сервера пошуку вузлів з подальшим вимкненням вузла. Для унеможливлення проведення подібних атак слід використовувати можливості середовища розгортки, математичні методи захисту інформації, передачу даних за допомогою брокерів повідомлень та двосторонню аутентифікацію об'єктів застосунку.

#### Список використаних джерел:

1. Распределенные системы. Принципы и парадигмы / Э. Таненбаум, М. Ван Стеен. – СПб. : Питер, 2003. – 877 с.
2. Распределенные приложения [Електронний ресурс]. – Режим доступу: [https://technet.microsoft.com/ru-ru/library/hh457612 \(v=sc.12\).aspx](https://technet.microsoft.com/ru-ru/library/hh457612 (v=sc.12).aspx).